TAMPEREEN
AMMATTIKORKEAKOULU

PELITUNTUMA JA SEN PARANTAMINEN VIDEOPELEISSÄ

Joonas Melanen

Opinnäytetyö
Huhtikuu 2018
Tietojenkäsittely
Pelituotanto

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietojenkäsittely
Pelituotanto

MELANEN, JOONAS:
Pelituntuma ja sen parantaminen videopeleissä

Opinnäytetyö 34 sivua, joista liitteitä 3 sivua
Huhtikuu 2018

Tässä opinnäytetyössä tutustuttiin videopelien kehityksen ja pelaamisen piilotettuun osaan, pelituntumaan. Opinnäytetyössä selvitettiin, mitä pelituntuma on, mistä se koostuu ja miten sitä voidaan parantaa. Opinnäytetyö toteutettiin osana Tampereen ammattikorkeakoulun tietojenkäsittelyn tradenomin tutkintoa pelituotannon suuntautumispolulla.

Opinnäytetyön tuloksena on tämä opinnäytetyöraportti. Raportissa kuvataan pelituntumaa ja sen eri elementtejä esimerkeillä ja kuvailemalla niiden vaikutusta pelaajaan. Pelituntuma terminä ja käsitteenä havaittiin melko subjektiiviseksi, mutta siitä voidaan silti löytää selvästi eriteltäviä osia, joita voidaan parantaa ja mitata enemmän tai vähemmän tarkasti. Näitä ovat esimerkiksi pelin syöte, vaste, konteksti, reaaliaikainen kontrolli, simuloitu tila ja viimeistely.

Pelin ja pelaajan välille muodostuu vuorovaikutteinen sykli, jossa pelaaja antaa komentoja pelille, joka vastaa näihin komentoihin ja antaa palautetta pelin muuttuneesta tilasta. Tämä sykli pitää sisällään kaiken sen, joka muodostaa tässä opinnäytetyössä esitetyn käsitteen pelituntumasta.

Jatkossa opinnäytetyössä kuvattuihin pelituntuman osiin voidaan keskittyä vielä entisestään. Jokaista pelituntuman osaa voidaan jatkossa tutkia ja analysoida perusteellisesti.

Asiasanat: pelituntuma, pelit, pelituotanto

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Business Information Systems
Game Development

MELANEN, JOONAS:
Game Feel and its Improvement in Video Games

Bachelor's thesis 34 pages, appendices 3 pages
April 2018

The subject of this thesis is to study the hidden part of designing and playing video games, game feel. The thesis clarifies what game feel is, what it is comprised of and how can it be improved. The thesis was carried out as part of Tampere University of Applied Science's Business Information Systems studies, specializing in Game Design.

The result of the thesis is this report. By using examples and describing its effect on the player, game feel as a term and a concept was found to be rather subjective. However, certain clearly definable parts that can be used to measure and improve game feel can be found. These include input, output, context, real-time control, simulated space and polish.

An interactive cycle is formed between the player and the game. The player gives a command to the game. The game then responds to this command and gives feedback on the changed state. This cycle contains all that forms the definition of game feel given in this thesis.

The analysis of the parts of game feel that are described in this thesis can be improved on by focusing on them individually. Every part of game feel can be examined and analyzed thoroughly, thus providing more extensive information on them.

Key words: game feel, games, game design

SISÄLLYS

LYHENTEET JA TERMIT

TAMK	Tampereen ammattikorkeakoulu
Unity	Pelien kehitykseen käytettävä pelimoottori
TV	Televisio
Perceptual field	”Aistikenttä”
ADSR	Attack, Delay, Sustain, Release – musiikkitermi

1 JOHDANTO

Idea tämän opinnäytetyön aiheesta syntyi koulussa tehdystä projektista. TAMKin pelituotannon suuntautumispolkuun kuului kolmiulotteisen pelin tekeminen Unity-pelimoottorilla. Pelistä tuli suunnitteluvaiheessa mekaniikoiltaan melko yksinkertainen, joten opettajan ehdotuksesta siitä pyrittiin tekemään mahdollisimman hyväntuntuinen. Tämän takia peliä kehittänyt ryhmä joutui myös selvittämään erilaisia tehokeinoja ja visuaalisia efektejä, joilla pelistä saatiin näyttävämpi ja viihdyttävämpi.

Videopelit voivat olla mekaniikoiltaan hyvinkin samanlaisia, mutta kun pelaaja istuu tietokoneensa eteen tai ottaa ohjaimen käteensä ja pelaa, hänen pelikokemuksensa voi olla viihdyttävyydeltään hyvinkin erilainen. Pelihahmo tuntuu liikkuvan joko liian liukkaasti tai hitaasti, pelin ohjaamiseen käytetyt kontrollit on jaoteltu ympäri ohjainta epäloogisesti ja ääniefektit ja musiikki ovat vaimeita. Pelit ovat mekaniikoiltaan samanlaiset, mutta ne tuntuvat erilaisilta.

Tämän pelituntuman eli ”game feelin” tutkiminen on tämän opinnäytetyön tarkoitus. Opinnäytetyössä määritellään mitä pelituntuma on, mistä se koostuu ja millaisin keinoin sitä voidaan mitata ja parantaa. Pelituntuman parantamiseen käytettäviä keinoja havainnollistetaan esimerkeillä edellä mainitusta TAMKin pelituotannon suuntautumispolulla tehdystä videopelistä, Sirius Ballsista. Tämä peli esitellään lyhyesti ennen pelituntumaan keskittymistä.

Opinnäytetyön tavoitteena on antaa aloitteleville pelinkehittäjille neuvoja heidän kehittämiensä pelien pelituntuman parantamiseen ja ohjata heitä tarkastelemaan pelejä eri tavoin.

2 SIRIUS BALLS

Sirius Balls tehtiin osana TAMKin Tietojenkäsittelyn koulutuksen Pelituotannon suuntautumispolkua. Peli tehtiin opintojen kolmantena vuotena kolmen hengen ryhmässä. Opinnäytetyön kirjoittaja oli yksi näistä kolmesta pienryhmän jäsenestä. Peliprojektin tarkoituksena oli tehdä kolmiulotteinen videopeli käyttäen Unity-pelimoottoria. Peli julkaistiin itch.io-verkkosivustolla ilmaiseksi joulukuussa 2016. Linkki pelin tuotesivulle on annettu liitteessä 1. Pelin alustana toimi Microsoftin Windows-käyttöjärjestelmä. Kuvassa 1 näkyy pelin aloitusruutu kokonaisuudessaan.

KUVA 1. Sirius Ballsin aloitusruutu

Sirius Balls on kahdelle hengelle tarkoitettu ja samaan aikaan pelattava kilpailupeli. Peli pohjautuu Taiton vuonna 1986 julkaisemaan Arkanoid-peliin, joka itse pohjautui Atarin vuonna 1976 julkaisemaan Breakout-peliin. Pelaajat käyttävät pelin pelaamiseen samaa näppäimistöä; heillä ei ole erillisiä ohjaimia. He myös pelaavat peliä jakaen saman näytön. Kuvassa 1 näytetystä aloitusruudusta pelaajat voivat oppia miten peliä pelataan (How to play), katsoa pelin tekijät (Credits), päättää voittoon vaadittavien maalien ja pisteiden määrän ja valita kentän.

Pelaajilla on kummallakin ohjattavanaan yksi maila ja alussa yksi pallo. Pelin tarkoituksena on tehdä valittu määrä maaleja tai saavuttaa annettu pistemäärä. Pisteitä saadaan tuhoamalla kentällä olevia erivärisiä palikoita tai tekemällä maaleja ohjaamalla oma

pallo vastustajan maaliin. Pallot kimpoavat mailoista, palikoista ja seinistä riippuen niiden osumakulmasta.

Kaikki palikat liikkuvat pelin aluksi paikoilleen liitteessä 2 olevan koodin avulla. Eri palikoilla on omat ominaisuutensa. Vaaleanpunaiset palikat antavat pelkästään pisteitä. Siniset palikat tekevät mailasta nopeamman, jotta pelaaja pysyy paremmin pallojen perässä. Vihreät palikat antavat sen rikkoneelle pelaajalle ylimääräisen pallon. Violetit palikat räjähtävät tuhoutuessaan, aiheuttaen vahinkoa niitä ympäröiville palikoille. Keltaiset palikat kasvattavat mailan kokoa. Palikoilla voi olla myös panssari, jolloin ne kestävät enemmän vahinkoa. Nämä palikat näkyvät kuvassa 2, jossa näkyy käynnissä oleva peli.

KUVA 2. Käynnissä oleva peli Sirius Ballsissa. Pelaaja 1 on keltainen, pelaaja 2 violetti

Peli on mekaniikoiltaan melko yksinkertainen, joten kehitystiimi keskittyi tekemään siitä mahdollisimman hyväntuntuisen. Tähän pyrittiin lisäämällä peliin visuaalisia efektejä ja muita tehokeinoja, jotka keskittyivät pääasiassa pelin ulkoasuun, mutta silläkin on oma osansa pelituntuman parantamisessa. Seuraavaksi opinnäytetyössä keskitytään itse pelituntumaan, käyttäen juuri kuvailtua Sirius Balls -peliä esimerkkinä havainnollistamaan pelituntuman parantamiseen käytettäviä keinoja.

3 PELITUNTUMA

3.1 Pelituntuman määritelmä

Itsenäisen pelinkehitysstudio Vlambeerin toinen avainjäsen, Jan Willem Nijman, sanoi tämän termistä ”game feel” Dutch Game Gardenin järjestämässä INDIGO-pelitapahtumassa vuonna 2013: ”That stupid word, feel; like, it’s a super abstract thing and it’s different for every player, but, you still have to work with it” (Youtube - Dutch Game Garden 2013). Nijmanin mielestä ”tuntuma” on liian abstrakti termi, minkä lisäksi se vaihtelee henkilöittäin. Jos heiltä kysytään ”mitä pelituntuma on”, jokainen voi antaa eri vastauksen.

Tässä opinnäytetyössä käytettävä määritys pelituntumasta tulee Steve Swinkin vuonna 2008 kirjoittamasta kirjasta *Game Feel: A Game Designer’s Guide to Virtual Sensation*. Kirjassaan Swink kuvaa pelituntuman koostuvan kolmesta eri osasta: ”real-time control”, reaaliaikaisesta kontrollista, ”simulated space”, simuloidusta tilasta ja ”polish”, viimeistelystä (Swink 2008, 2).

3.1.1 ”Real-time control” eli reaaliaikainen kontrolli

Videopelien reaaliaikaisessa kontrollissa on kaksi osallistujaa, jotka tässä tapauksessa ovat pelaaja ja hänen pelaamiseen käyttämänsä laite, kuten tietokone, pelikonsoli tai puhelin. Käyttäjä ja laite muodostavat keskenään silmukan, jossa käyttäjällä on tavoite, jonka täyttääkseen hän antaa laitteelle komennon esimerkiksi painamalla ohjaimen painiketta. Laite vastaanottaa tämän komennon, käsittelee sen, ja tulostaa sen aiheuttaman muutoksen. Käyttäjä havaitsee tämän muutoksen, vertaa sitä alkuperäiseen tavoitteeseensa ja antaa sitten laitteelle uuden komennon.

Keskustelun sijaan Swink vertaa tätä silmukkaa autolla ajamiseen, jossa toiminta tulee selkäytimestä ja reaktiot muutoksiin ovat nopeita. Tämän vuorovaikutuksen tulee myös olla keskeytymätöntä. Pelissä on myös oltava objekti, jota ohjata ja joka reagoi pelaajan antamiin komentoihin (Swink 2008, 3).

3.1.2 ”Simulated space” eli simuloitu tila

Reaaliaikainen kontrolli on kuitenkin turhaa, jos se tapahtuu tyhjiössä. Kuinka pelihahmon hypyn korkeutta tai nopeutta voi arvioida, jos sen ympärillä ei ole mitään mihin verrata sen liikettä? Tämän takia tarvitaan myös simuloitu tila.

Simuloitu tila on pelihahmon ympäristö. Tämä ympäristö pitää sisällään pelissä olevat objektit, joiden kanssa pelihahmo tulee toimimaan ja tästä toiminnasta seuraavat seuraukset. Myös kenttäsuunnittelu kuuluu simuloituun tilaan. Nämä objektit ja ympäristöt antavat merkityksen pelihahmon liikkeelle. Pelaajalle peliympäristö toimii tilaisuutena havaita ja ilmaista itseään liikkumalla, tutkimalla ja koittamalla sen rajoja. Simuloidun tilan tulisi myös olla aktiivisesti koettavissa. Elokuvat ja TV ovat passiivista kokemista, mutta videopeleissä pelaaja kokee ympäristön itse omalla tahdillaan (Swink 2008, 4-5).

3.1.3 ”Polish” eli viimeistely

Tällä hetkellä pelituntumaa kuvaa virtuaalinen objekti, jota ohjataan reaaliaikaisesti simuloidussa tilassa. Mutta tämä tila ja pelihahmo voisivat olla pelkkiä sopivan kokoisia palikoita ja peli silti toimisi mekaanisesti samalla tavalla. Tässä vaiheessa mukaan tulee viimeistely.

Viimeistelyssä peliin lisätään muun muassa audiovisuaalisia efektejä, jotka vahvistavat pelaajan vuorovaikutusta pelin kanssa vaikuttamatta sen toimintaan. Näitä efektejä ovat esimerkiksi kipinä miekan osuessa kilpeen, lukon avautumisen ääniefekti ja animaatio hahmon hypystä. Näillä efekteillä pelistä pyritään tekemään miellyttävämpi katsoa ja kokea. Esimerkkejä näistä efekteistä käsitellään myöhemmin opinnäytetyön luvussa 4.4, ”Polish” eli viimeistely.

Viimeistelyn jälkeen Swinkin määritelmä pelituntumasta on valmis: virtuaalinen objekti, jota ohjataan reaaliaikaisesti simuloidussa tilassa, ja jonka vuorovaikutusta ympäristön kanssa vahvistetaan viimeistelyllä. Pelaaja ohjaa pelihahmoa, pelihahmo on vuorovaikutuksessa ympäristönsä kanssa ja viimeistelyn efektit vahvistavat tätä vuorovaikutusta ja tekevät pelistä miellyttävämmän (Swink 2008, 6).

3.2 Pelituntuma kokemuksina

Pelituntuma pelaajan kokemana tuntemuksena koostuu monista eri kokemuksista. Näitä voivat olla ohjaamisen ja pelimekaniikan hallinta, haasteiden ylittäminen ja uusien kokemusten löytäminen. Kirjassaan Swink (2008, 10) on esittänyt muutamia tavallisimmista pelituntuman esiintymistä: hallinnan esteettinen tunne, uuden taidon oppiminen, harjoittelu ja hallinta, aistien ja identiteetin laajentuminen ja vuorovaikutus pelimaailmassa.

3.2.1 Hallinnan esteettinen tunne

Hallinnan esteettisessä tunteessa pelkkä pelihahmon ohjaaminen on suuri osa pelituntumaa. Pelihahmo seuraa pelaajan komentoja juuri oikein ja ilman viivettä. Pelaaja ei edes tarvitse mitään tavoitetta nauttiakseen pelaamisesta. Tätä tunnetta pelaajat tarkoittavat kuvaillessaan pelin kontrollien olevan sulavia, leijuvia tai jäykkiä (Swink 2008, 14).

Hyvä esimerkki tästä ovat ajopelit. Ajopeleissä pelaaja ajaa esimerkiksi autoa tai muuta kulkuvälinettä. Tavoitteena on usein kisata muita pelaajia tai tietokoneen ohjaamia vastustajia vastaan tai suorittaa annettu reitti mahdollisimman nopeasti. Näille ajopeleille ehdottoman tärkeää on hyvän ajotuntuman saavuttaminen: kulkuväline kääntyy pelaajan määräämään suuntaan tarkasti, kaasuttaminen ja jarruttaminen ovat sopivan joustavia ja pelin antama palaute, oli se sitten audiovisuaalista tai ohjaimen välittämää, vastaa pelaajan odottamaa reaktiota ja auttaa häntä valitsemaan seuraavan liikkeensä. Pelaaja uppoutuu ajamiseen ja voi jopa itse liikkua pelihahmonsa mukana, vaikkei sillä ole mitään vaikutusta.

Tätä tunnetta luodessaan pelinkehittäjän täytyy ensin päättää, millä syötteellä liike tapahtuu ja mikä vaikutus sillä on liikkeeseen. Pelaajan painaessa ohjaimen painiketta vaste voi olla välitön tai se voi tapahtua hiljalleen. Hahmo voi liikkua suoraan ohjaimen sauvan osoittamaan suuntaan tai kääntyä sitä kohti. Itse ohjaamiseen käytettävä ohjain on myös otettava huomioon: Sonyn Playstation 4-konsolin DualShock 4 -peliohjaimen liipaisimet ovat paljon sopivampia kaasuttamiseen kuin sen muut painikkeet. Kun syöte

ja vaste ovat harmoniassa, pelin ohjaamisesta tulee luonnollisen helppoa ja tyydyttävää (Swink 2008, 14).

3.2.2 Uuden taidon oppiminen

Pelaaja voi siis nauttia pelkästä pelihahmon ohjaamisesta tai ympäristön tutkimisesta, mutta jossain kohtaa hän tulee kohtaamaan jotakin uutta, joka haastaa hänen tämänhetkisen hallintansa pelin kontrolleista ja mekaniikoista. Tasoloikkapelissä kyseessä voisi olla totuttua leveämpi rotko, jota pelaaja ei voi ylittää tähän saakka käyttämällään hypyllä. Hiiviskelypelissä, jossa tavoitteena on pysyä piilossa vastustajilta, vastustajat saattavat alkaa kuulla pelaajan tämän juostessa äänekkäästi paikasta toiseen. Peli esittää haasteen, joka pelaajan on opittava ylittämään.

Tämä haaste ohjaa pelaajaa tutkimaan hänelle tarjolla olevia mahdollisuuksia. Yhdistämällä aiemmin oppimaansa tai kokeilemalla pelin tarjoamien mekaniikoiden rajoja pelaaja kehittää uusia taitoja hahmonsa ohjaamiseen ja pelin hallitsemiseen. Pitämällä yhtä painiketta pohjassa pelihahmo kiihdyttää vauhtiaan, ja painamalla aiemmin opittua hyppäämiseen käyttämää painiketta pidempään pelihahmo hyppää korkeammalle. Näiden yhdistäminen mahdollistaa aiemmin ylitsepääsemättömän esteen ylittämisen.

Tämä ahaa-elämys on tärkeä osa pelituntumaa. Pelaaja on saavuttanut syvemmän pelin kontrollien hallinnan tason, mikä mahdollistaa pelin tarjoamien haasteiden ylittämisen uusilla keinoilla. Peli palkitsee pelaajan kokeilemisen tarjoamalla uusia mahdollisuuksia näiden uusien taitojen soveltamiseen ja ohjaa pelaajaa koettelemaan rajojaan esittämällä hänelle uusia haasteita. Pelituntuma muuttuu pelaajan taitojen kasvaessa (Swink 2008, 16).

3.2.3 Harjoittelu ja hallinta

Pelaaja on nyt siis oppinut uuden taidon, joka mahdollistaa pelin pelaamisen ja kokemisen entistä laajemmilla keinoilla. Tämä uusi taito ei kuitenkaan ole välttämättä välittömästi täysin hallittavissa. Taistelupelissä pelaajan hahmolla voi olla erikoisliike, jonka toteuttaminen vaatii tarkan ja mahdollisesti monimutkaisen painikeyhdistelmän, jota

pelaaja ei vielä saa luotettavasti toistettua. Kyseessä voi myös olla pelin esittämä haaste, joka vaatii tarkkaa suorittamista. Kuten minkä tahansa uuden oppiminen, tämä vaatii harjoittelua.

Pelaaja asettaa itselleen tavoitteen, joka ohjaa hänen toimintaansa ja pelikokemusta. Tässä tapauksessa pelaajan tavoitteena olisi oppia suorittamaan erikoisliikkeen vaatima näppäinyhdistelmä luotettavasti. Harjoitellessaan tämän tavoitteen saavuttamista pelaaja saattaa myös löytää uusia menetelmiä ja oppia uusia taitoja, joilla hän voi laajentaa pelikokemustaan. Lopulta pelaajalla on laajempi ja varmempi ote pelin mekaniikoista ja hän tuntee hallitsevansa pelin.

3.2.4 Aistien ja identiteetin laajentuminen

Kirjassaan Swink (2008, 25) esittää seuraavan väittämän pelituntumasta aistien laajenemisena: ”To experience game feel is to see through different eyes, hear through different ears and touch with a different body”. Keskittyessään peliin pelaaja sulkee muun maailman pois ja hänen aistinsa tavallaan siirtyvät peliin. Monitori tai televisio toimii ikkunana pelimaailmaan, jonka kautta pelaaja kokee pelimaailman omana, väliaikaisena ympäristönään. Kaiuttimet tai kuulokkeet välittävät peliympäristön äänet pelaajalle hänen hahmonsa kautta: pelaaja itse kuulee uhkaavan äänen takaansa ja tietää täten kääntyä. Pelin ohjaamiseen käytettävä ohjain, on se sitten hiiri ja näppäimistö, konsoliohjain tai VR -liikeohjaimet, tärisee tai muuten antaa fyysistä palautetta.

Jokaisessa tapauksessa pelin vaste korvaa jonkin pelaajan edellä kuvatuista kolmesta aistista: pelin ulkoasu ja visuaalinen palaute ruudun kautta korvaa näön, musiikki ja ääniefektit sekä niiden sijoittaminen pelimaailmaan kuulon ja ohjaimen tärinä tai liikuttaminen tunnon. Tämä aistien laajentaminen auttaa pelaajaa reagoimaan peliin paremmin ja uppoutumaan siihen.

Myös pelaajan identiteetti voi laajentua peliin: pelaaja kokee pelin tapahtumien tapahtuvan hänelle itselleen. Swink vertaa tätä autolla ajamiseen: kuljettaja laajentaa havaintoaan kattamaan koko auton ja sen ympäristön, jolloin autosta tulee kuljettajan jatke Auton osuessa tolppaan kuljettaja itse irvistää, ikään kuin häneen itseensä olisi sattunut

(Swink 2008). Tilanne on sama pelien kanssa, ainakin jos niissä ohjataan pelihahmoa suoraan.

Pelaaja on hahmonsa kautta suorassa vuorovaikutuksessa pelimaailman kanssa. Täten hän itse tavallaan vaikuttaa pelimaailmaan ja reagoi pelin vasteeseen tästä vuorovaikutuksesta. Pelinkehittäjä Jonathan Blow, joka on tunnettu peleistään Braid (2008) ja The Witness (2016) kutsuu tätä termillä ”proxied embodiment” (Varney 2008) eli edustava ruumiillistuma. Osuessaan vastustajaansa ensimmäisen persoonan ammuntapelissä pelaaja voi huudahtaa ”jes, osuin!” eikä ”jee, pelihahmoni osui!”. Vastineeksi ollessaan liian hidas väistämään hän saattaa valittaa ”miten se ei nyt ehtinyt väistää”.

3.2.5 Vuorovaikutus pelimaailmassa

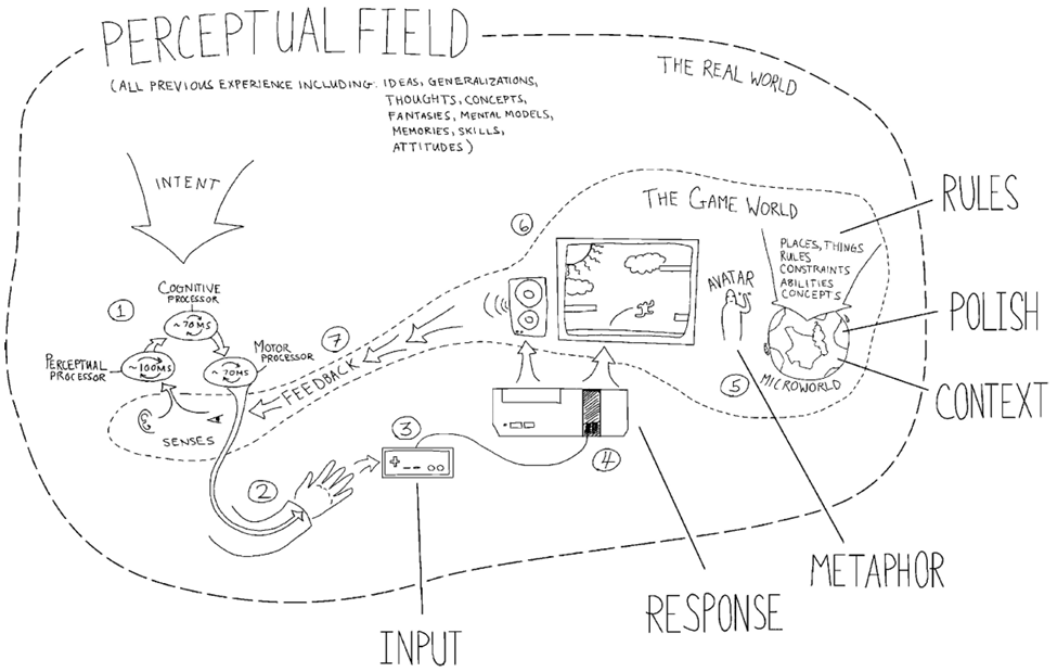
Pelaaja tuo mukanaan peliin myös tiettyjä olettamuksia sen käyttäytymisestä ja miten pelimaailma toimii. Nämä olettamukset tulevat hänen aiemmista kokemuksistaan ja tietämyksestään joko samanlaisista peleistä tai oikeasta maailmasta. Pelatessaan realistista urheilupeliä pelaaja olettaa jalkapallon kimpoavan tolpasta riippuen sen tulokulmasta ja vauhdista, koska niin hän olettaa jalkapallon toimivan. Pelaaja tietää pelin seuraavan edes lähes todellisia fysiikan lakeja pallon reagoidessa kuten sen pitäisi. Tämän ansiosta hän voi olettaa tiettyjen toimintojen johtavan odotettuun lopputulokseen.

Pelien tavoittelemasta realismista huolimatta videopelien fysiikat ovat lähinnä imitaatioita oikean maailman vastineista. Ihmiset ovat kuitenkin hyviä arvioimaan ja päättelemään virtuaalisen ympäristön fysiikoita juuri koska he tietävät miten niiden tulisi oikeasti toimia. Samalla pelin audiovisuaalinen tyyli voi vaikuttaa pelaajan olettamuksiin. Pelaajat päättelevät ja oppivat pelimaailman vihjeistä ja reaktioista, miten kyseinen maailma toimii (Swink 2008, 30).

Pelin visuaalisen tyylin ja sen fysiikoiden ollessa harmoniassa pelaaja voi paremmin uppoutua pelimaailmaan ja se tuntuu paremmalta. Näiden elementtien ollessa ristiriidassa tämä illuusio kuitenkin särkyy: peli voi olla ulkoasultaan ultrarealistinen, mutta pelaajan kuvitelma pelin maailmasta särkyy heti kun jokin pelimaailmassa ei käyttäydy ”oikein”.

3.3 Pelituntuman vuorovaikutteisuus

Tässä vaiheessa on tarkennettu pelituntuman koostuvan kolmesta osasta, jotka yhdessä muodostavat seuraavan määritelmän: pelituntuma koostuu virtuaalisesta objektista, jota ohjataan reaaliaikaisesti simuloidussa tilassa, ja jonka vuorovaikutusta ympäristön kanssa vahvistetaan viimeistelyllä. Lisäksi pelituntuma pelaajan tuntemuksina koostuu monista eri kokemuksista, joista muutama esiteltiin kappaleessa 3.2. Pelituntuma on siis pelaajan ja pelin välinen, vuorovaikutteinen sykli. Swink (2008, 62) on kirjassaan kuvannut tämän syklin vaihe vaiheelta (kuva 3).



KUVA 3. Swinkin kaavio pelituntuman vuorovaikutuksen syklistä (Kuva: Steve Swink 2008)

Swinkin syklissä kaikki pelituntuman syklissä sisältyy ja saa alkunsa ”perceptual fieldistä”. Tämä pitää sisällään kaikki pelaajan aikaisemmat kokemukset, ideat, ajatukset, muistot ja muut konseptit, jotka muodostavat hänen kuvansa objektiivisestä todellisuudesta (Swink 2008, 61). Tästä syntyy pelaajan tavoite, jonka hän haluaa toteutuvan.

Tästä tavoitteesta lähtee liikkeelle pelituntuman vuorovaikutuksen seitsenosainen sykli.

1. ”Human processor”, ihmisprosessori
2. Lihakset

3. Syötelaite (pelin ohjaamiseen käytetty ohjaustapa)
4. Tietokone, joka käsittelee pelaajan antamat komennot
5. Pelimaailma
6. Vastelaite (TV, Monitori)
7. Aistit

Pelaajalla on tavoite (”intent”), jonka hän haluaa toteuttaa. Ihmisprosessorin ensimmäinen osa, ”cognitive processor” eli kognitiivinen prosessori, käsittelee tämän tavoitteen ja välittää sen ”motor processorille” eli liikeprosessorille. Liikeprosessori ohjaa pelaajan lihaksia toteuttamaan tämän tavoitteen painamalla ohjaimen painikkeita ja antamaan pelille komentoja. ”Perceptual processor” eli aistiprosessori ottaa vastaan pelin antaman vasteen ja välittää sen kognitiiviselle prosessorille käsittelyä varten. Nämä kolme prosessorin osaa muodostavat yhdessä oman syklinsä pelituntuman vuorovaikutuksessa.

Kuvassa 3 kohdassa 1 näkyvät ajat millisekunneissa ovat keskiarvoja ajasta, joka tavallisella ihmisellä kestää välittää uutta tietoa eri prosessoreille. Nämä ajat ovat 70 millisekuntia kognitiiviselle, 70 ms liikkeelle ja 100 ms aisteille (Swink 2008, 36). Aistit saavat ärsykkeen (jokin muuttuu pelissä), joka käsitellään ja sitä seuraava komento lähetetään lihaksille. Lihakset vastaanottavat viestin aivoilta ja toteuttavat pelaajan antaman komennon.

Syötelaite välittää pelaajan tavoitteen tietokoneelle. Pelaajan tavoite voi olla monimutkainen tai epäselvä, joten se täytyy välittää tietokoneelle sen ymmärtämässä muodossa. Pelaaja painaa ohjaimen painikkeita, kääntää ohjaussauvaa tai liikuttaa hiirtä oikeaan suuntaan. Kappaleessa 4.1 kerrotaan lisää syötteestä ja sen vaikutuksesta pelituntumaan.

Ohjain välittää pelaajan komennon tietokoneelle, joka sitten käsittelee sen ja päivittää peliä komentojen mukaan. Pelin ulkoasun, äänitehosteiden ja tilan täytyy päivittyä niin nopeasti, että pelaaja kokee pelin vasteen olevan välitön. Tämän kaiken täytyy tapahtua yhden aistiprosessorin syklin (vähimmillään 50ms, keskiarvo 100ms) aikana. Peli tuntuu sulavalta ja keskeytyksettömältä tietokoneen käsitellessä komennon nopeammin kuin pelaaja ehtii reagoida siihen (Swink 2008, 64). Kappaleessa 4.2 kerrotaan enemmän tietokoneen puolella tapahtuvista laskennoista ja sen antamasta vasteesta.

Pelimaailmoja on pelissä tavallaan kaksi kappaletta. Toinen on tietokoneen sisällä tapahtuva sarja laskutoimituksia, komentoja ja ehtolausekkeita. Toinen on pelaajan ko-

kema, tyylitellympi versio tästä maailmasta, jonka hän vastaanottaa vastelaitteiden, kuten monitorin ja kaiuttimien, kautta. Pelimaailmassa on myös omat, selvästi määritellyt tavoitteensa ja maalinsa, jotka ohjaavat pelaajan toimintaa. Pelaaja kokee tämän pelimaailman pelihahmonsa kautta, oppien lisää sen toiminnasta, mekaniikoista, vaatimista taidoista ja haasteista.

Vastelaitteet, kuten TV tai monitori, kaiuttimet ja kuulokkeet ja jopa peliohjaimen värinä, ovat pelaajan ikkuna pelimaailmaan (Swink 2008, 65). Nämä laitteet esittävät tietokoneen sisäisten laskujen tuloksen pelaajalle ymmärrettävässä muodossa. Pelissä on saavutettu uusi tila, joka välitetään pelaajalle vastelaitteiden kautta visuaalisesti, äänenä ja jopa käsinkosketeltavana tuntemuksena.

Viimeisessä vaiheessa pelaajan aistit ottavat vastaan pelin päivitetyn tilan. Pelaaja näkee, kuulee ja tuntee tilanteen muuttuneen. Nämä aistimukset päätyvät aistiprosessorin kautta kognitiiviselle prosessorille, joka käsittelee ne uuden syklin aloittamista varten.

4 PELITUNTUMAN PARANTAMINEN

Kuten on käynyt selväksi, pelituntuma on hyvinkin abstrakti ja epämääräinen, henkilökohtainen tunne. Joissakin tapauksissa yksi pelaaja voi kokea pelin kontrollien olevan aivan liian epätarkkoja, kun taas toinen nauttii tästä aiheutuvasta vapaammasta, liukkaasta tunteesta. Jokaisella voi olla oma määritelmänsä sille, mitä on hyvä pelituntuma. Jotta kehittäjä voi luotettavasti ja varmasti tehdä mahdollisimman hyvältä tuntuvan pelin, on pelituntumalle kuitenkin löydettävä joitakin enemmän tai vähemmän tarkasti mitattavissa olevia osia, joita voidaan selkeästi parantaa.

Koska pelituntuma on pelaajan tuntema ja kokema tunne, on sen ja sen osien mittaaminen mahdollista ainoastaan tarkkailemalla pelaajia ja heidän kokemuksiaan pelin parissa. Tätä varten pelien kehittäjät voivat järjestää avoimia tai suljettuja pelitestejä pelin kehityksen aikana. Tietty osa tai keskeneräinen versio pelistä annetaan pelin oletetun yleisön pelattavaksi, minkä aikana pelaajien reaktioita ja tilastoja mittaamalla saadaan tarkasti mitattavissa olevaa tietoa pelaajien käyttäytymisestä ja käyttötarinoita pelaajien kokemuksista pelin parissa.

Tässä luvussa käydään läpi näitä pelituntuman osia, joista on mahdollista tehdä objektiivisesti parempia. Jokaista osaa havainnollistetaan käyttämällä opinnäytetyön luvussa 2 esiteltyä videopeliä Sirius Balls esimerkkinä. Peliä ja sen toteutusta verrataan käsiteltävänä olevaan osaan ja jokaisesta kohdasta esitetään myös mahdollisia jatkokehitysehdotuksia. Tavoitteena on esittää, miten kyseinen pelituntuman parantamiseen käytetty osa toteutettiin Sirius Ballsissa, ja miten sitä olisi voinut vielä parantaa.

4.1 Kehittäjän hallussa olevat pelituntuman osat

Kappaleessa 3.3 kuvatussa pelituntuman vuorovaikutteisessa syklissä on erikseen merkitty kuusi termiä isoilla kirjaimilla: ”Input” eli syöte, ”Response” eli vaste, ”Context” eli konteksti, ”Polish” eli viimeistely, ”Metaphor” eli metafora ja ”Rules” eli säännöt. Nämä ovat pelituntuman osia, joihin pelin kehittäjät voivat vaikuttaa. Pelin ohjaamiseen käytettävä ohjain ja pelin kontrollien herkkyys, pelaajan antamien komentojen vasteai-

ka, pelimaailman tavoitteet ja ympäristöt sekä pelin antama audiovisuaalinen palaute ovat esimerkkejä näistä (Swink 2008, 85).

Seuraavissa luvuissa keskitytään kerrallaan kaikkiin näihin kuuteen osaan ja annetaan esimerkki niiden toteutuksesta Sirius Ballsissa. Lisäksi jokaisessa kappaleessa pohditaan uusia tapoja ja keinoja Sirius Ballsin pelituntuman parantamiseksi. Kappaleissa annetaan myös mittareita ja esimerkkejä pelituntuman osien, kuten esimerkiksi syötteen, mittaamista varten.

4.2 ”Input” eli syöte

Syöte on pelin ja pelaajan välissä oleva fyysinen rajapinta, jonka kautta pelaajan tavoite välittyy peliä suorittavalle laitteelle. Kyseessä voi olla pelikonsolin ohjain, tietokoneen hiiri ja näppäimistö, puhelimen tai tabletin kosketusnäyttö tai mikä tahansa muu lukuisista pelien lisävarusteista, kuten ratti ja polkimet ajopelejä varten. Kehittäjän on ymmärrettävä pelin ohjaamiseen käytettävä laite tai ohjain, mitä sillä voi tehdä ja mitä kaikkia toimintoja siinä on.

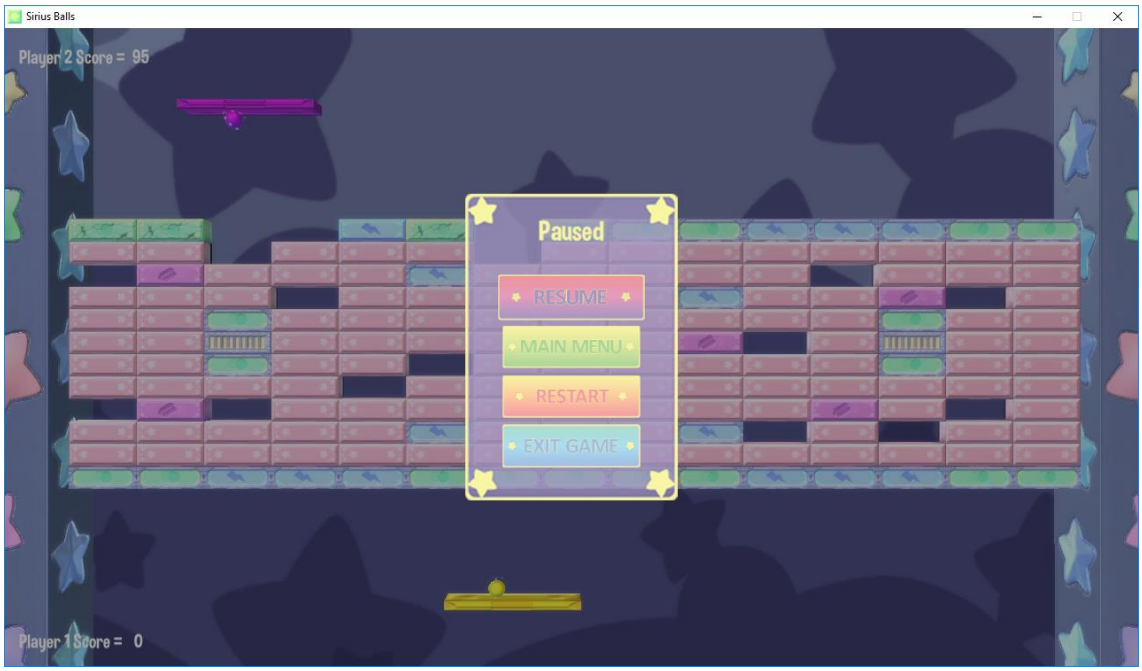
Pelin ohjaamisen käytetyn laitteen ymmärtäminen helpottuu jakamalla se kolmeen eri tasoon: mikro-, makro- ja taktiilinen taso. Mikrotasolla tutkitaan syötelaitteen, kuten tietokoneen näppäimistön, jokaista käytettävissä olevaa painiketta tai toimintoa, jolla on mahdollista ohjata peliä tai antaa sille komentoja. Makrotasolla keskitytään syötelaitteen mahdollisuuksiin: mitä kaikkea tällä laitteella voidaan tehdä, millainen ohjain on kokonaisuutena, minkä kaltaisiin peleihin tai tilanteisiin se on sopiva ja niin edelleen. Taktiilisella tasolla kiinnitetään huomiota syötelaitteen rakenteeseen, painikkeiden sijoitteluun ja miltä kyseinen pelin ohjaamiseen käytettävä laite tuntuu pelaajan käsissä. Tähän pelin kehittäjä ei usein voi vaikuttaa, vaan se riippuu pelin alustan (pelikonsoli, tietokone ja niin edelleen) yleisistä syötelaitteista, kuten pelikonsolien ohjaimista. (Swink 2008, 102).

Jokaiselle syötelaitteelle yhteistä on mahdollisuus liikkeeseen tiettyjen rajoitteiden mukaan. Näppäin painetaan pohjaan, jolloin sillä on ainoastaan kaksi eri tilaa: pohjassa tai ylhäällä. Pelikonsolien ohjaimien liipaisin voidaan myös painaa pohjaan, mutta sillä on useita muita tiloja ennen pohjan saavuttamista. Tietokoneen hiirtä liikutetaan vasem-

malta oikealle ja ylös ja alas tasaisella pinnalla. Hiiren rajoitteina ovat ainoastaan näytön rajat ja hiiren tasona toimivan alueen pinta-ala. Virtuaalilasien ohjainta voidaan liikuttaa täysin kolmiulotteisesti. Tietokone tulkitsee tämän syötteen ja antaa pelaajalle vastelaitteiden kautta palautetta syötteen tuloksesta. Syötteen liike voidaan myös jakaa jatkuvaan ja välittömään liikkeeseen. Esimerkiksi hiiri päivittää sijaintiaan tietokoneelle jatkuvasti, kun taas näppäintä painaessa tietoa lähetetään tietokoneelle välittömästi, vain silloin kun näppäintä painetaan. (Swink 2008, 103).

Sirius Ballsia ohjataan tietokoneen näppäimistöllä ja hiirellä. Useimmissa näppäimistöissä on 104 näppäintä, mutta tämä numero voi vaihdella valmistajasta tai näppäimistön mallista riippuen (Computer Hope 2017). Tietokoneen hiiressä on näppäimiä yleensä kolme, mutta näppäimien määrä voi vaihdella yhdestä jopa yli kymmeneen.

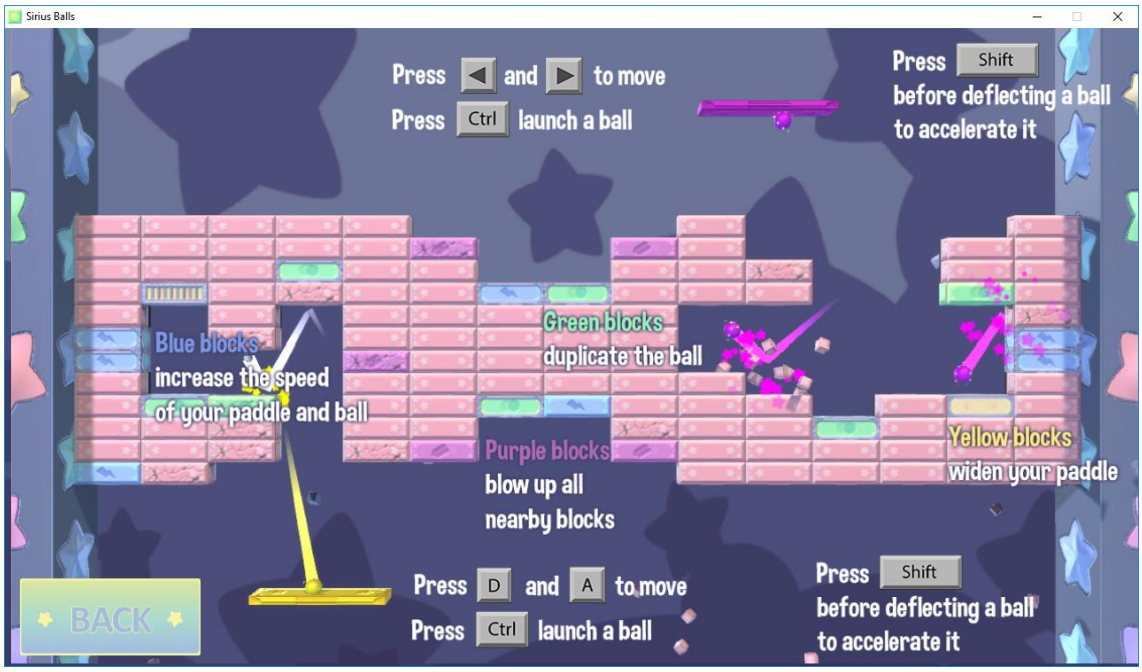
Sirius Ballsissa hiirtä käytetään ainoastaan pelin valikoissa. Ennen pelin alkua hiirellä määritetään voittoon vaadittavien maalien ja pisteiden määrä ja valitaan pelikenttä. Hiirtä ei käytetä mailojen tai pallon ohjaamiseen. Jos toinen pelaajista pysäyttää pelin, hiirtä käytetään kuvassa 4 näkyvien pelin valikoiden selaamiseen. Valikoista voidaan palata peliin tai päävalikkoon, käynnistää kyseinen ottelu uudestaan tai sulkea peli.



KUVA 4. Valikot pelin aikana.

Itse peliä ohjataan näppäimistöllä. Pelaaja 1 (keltainen pallo ja maila) ohjaa mailaansa näppäimistön A- ja D-näppäimillä, ampuu pallon liikkeelle näppäimistön vasemman-

puoleisella Ctrl-näppäimellä ja kimmottaa pallon tehokkaammin painamalla vasemmanpuoleista Shift-näppäintä. Pelaaja 2 (violetti pallo ja maila) ohjaa mailaansa näppäimistön nuolinäppäimillä (vasen ja oikea) ja käyttää muihin toimintoihin näppäimistön oikeanpuoleisia vastineita pelaaja 1:n kontrolleista. Molemmat pelaajat voivat keskeyttää pelin ja avata valikon painamalla näppäimistön Esc-näppäintä. Peliin voi palata joko hiirellä tai painamalla Esciä uudestaan. Pelin ohjaamiseen käytettävät kontrollit näkyvät kuvassa 5. Hiirtä lukuun ottamatta pelin kontrollit ovat välittömiä.



KUVA 5. Pelin ohjaamiseen käytettävät kontrollit

Kun syötelaite on jaettu osiin, voidaan keskittyä laitteeseen yhtenä kokonaisuutena ja tarkastella, mitä kaikkea sillä voidaan tehdä. Tätä varten kehittäjä voi tehdä listan kaikista syötelaitteen painikkeista ja syötteistä ja niiden mahdollisista yhdistelmistä ja rajoitteista. Kun kehittäjä tietää kaiken, mitä syötelaitteella on mahdollista tehdä, ja tietää mitä hän haluaa pelaajan pystyvän pelissä tekemään, hän voi suunnitella pelin kontrollit.

Sirius Ballsin yhdellä pelaajalla on käytössään neljä näppäintä näppäimistöstä: vasemmalle ja oikealle liikkumiseen tarkoitetut ja pallon laukaisua ja kimmottamista varten tarkoitetut näppäimet. Kaikilla näppäimillä on vain kaksi tilaa: ylhäällä ja pohjassa. Lisäksi liikkumiseen tarkoitetut näppäimet sulkevat toisensa pois: pelaaja ei voi liikkua samaan aikaan sekä vasemmalle että oikealle. Pelaaja voi siis liikkua joko vasemmalle tai oikealle ja samaan aikaan ampua pallon tai kimmottaa sen.

Sirius Ballsin pelaajalla on melko vähän mahdollisuuksia soveltaa pelin kontrolleja, koska eri toimintoja ei ole useita ja kaikilla näppäimillä on vähän mahdollisia tiloja. Samalla pelin yksinkertaiset kontrollit ja mekaniikat tekevät pelistä helpon pelata aloittelijallekin: pelaajan ei tarvitse osata hienoja tai tarkkoja ohjausliikkeitä pärjätäkseen pelissä. Pelaajat jakavat saman näppäimistön, minkä seurauksena kontrollit on sijoitettava tarpeeksi erilleen toisistaan. Näin pelaajat saavat mahdollisimman paljon tilaa pelata eivätkä jää toistensa tielle.

Tiettyjen pelin ohjaamiseen tarvittavien näppäimien käyttäminen voidaan johtaa pelaajien tottumuksista: nuolinäppäimet ja WASD-kirjainnäppäimet ovat yleistyneet pelihahmon ja valikoiden liikuttamiseen (Wilde 2016). Sen lisäksi että kehittäjä tietää, mitä kaikkea syötelaitteella voidaan tehdä, on hänen myös hyvä ymmärtää, mitä pelaajat olettavat syötelaitteella olevan mahdollista tehdä ja miten se yleensä toimii.

Tietyt pelien kontrolliskeemat ovat muodostuneet alan standardeiksi: ensimmäisen persoonan kuvakulmasta kuvatussa sotapelissä hiiren vasen näppäin ampuu ja oikea tähtää, ajopelissä pelikonsolin ohjaimen oikea liipaisin on kaasu ja vasen jarru ja niin edelleen. Pelaajat ovat tottuneet näihin kontrolleihin muista genren peleistä, joten kehittäjä voi niitä käyttämällä vähentää pelin kontrollien oppimiseen vaadittavaa aikaa. Päinvastaisesti kehittäjä voi myös tehdä peliinsä täysin omalaatuisen kontrolliskeeman. Tämä vaatii lisää työtä kehittäjältä, mutta sen ansiosta peli voi tuntua uniikilta ja mielenkiintoiselta (Salmond 2016, 120).

Sirius Ballsin ohjaamiseen tarvitaan tällä hetkellä sekä hiirtä että näppäimistöä. Tässä tapauksessa pelin syötteestä voitaisiin tehdä yksinkertaisempaa ja vaivattomampaa ottamalla hiiri kokonaan pois kontrolleista. Jos koko peliä (sen valikot, kenttien valitseminen, pelin sammuttaminen jne.) olisi mahdollista ohjata näppäimistöllä, pelaajien ei tarvitsisi näitä toimintoja varten irrottaa käsiään näppäimistöstä. Tämä myös vähentäisi pelin pelaamiseen vaadittavaa tilaa: hiirelle ei ole tarvetta löytää tasaista pintaa (kannettavalla tietokoneella tätä ongelmaa ei kosketuslevyn ansiosta olisi). Peliin ei edes täytyisi lisätä ylimääräisiä graafisia efektejä osoittamaan, mikä pelin painikkeista on valittuna: tällä hetkellä valittu painike kasvaa koossa, kun hiiri on sen päällä.

4.3 ”Response” eli vaste

Vaste on tietokoneen tai muun pelin pelaamiseen käytettävän laitteen antama palaute pelaajan antamasta komennosta. Pelaaja antaa komennon laitteelle syötelaitteen kautta, minkä jälkeen laite tulkitsee signaalin ja muuttaa jotain pelissä sen perusteella. Tämä voi olla pelihahmon sijainti pelimaailmassa, sen nopeus, sen koko tai pelihahmon tila. Vasteena pelaajan komentoon peli voi myös luoda kokonaan uuden objektin pelimaailmaan tai toistaa äänileikkeen (Swink 2008, 119).

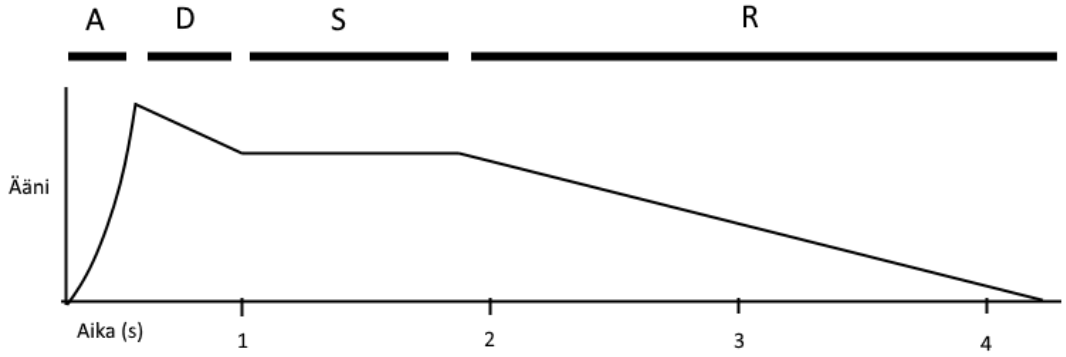
Pelaajan antama signaalin tyyppi riippuu sen antamiseen käytettävästä syötelaitteesta. Esimerkiksi tietokoneen näppäimellä voi olla vain kaksi tilaa, ylhäällä tai pohjassa. Tällöin näppäimen antama signaali on samalla tavalla muotoa päälle tai pois; sillä ei voi olla muita tiloja näiden lisäksi eikä se siten voi lähettää muunlaisia signaaleja. Tietokoneen hiiri sen sijaan lähettää jatkuvasti sijaintiaan, minkä muutokset se lähettää X- ja Y-akselin koordinaatteina.

Pelaajan antaman signaalin luonteesta riippumatta sillä on jokin vaikutus pelissä. Tämän vaikutuksen nopeutta ja kestoa muokkaamalla voidaan vaikuttaa pelituntumaan. Jos pelaajan tarkoituksena esimerkiksi on liikuttaa pelihahmoa oikealle, hän painaa tätä liikettä vastaavan näppäimen pohjaan (näppäimen sijaan kyseessä voi olla peliohjaimen sauva tai muu vastaava ohjaamiseen käytettävä syöte). Ohjain lähettää signaalin pelille, joka muokkaa hahmon liikuttamista ohjaavaa arvoa sen mukaan. Tämän arvon kasvua ja muutosta voidaan ohjata niin kutsutun ADSR-verhokäyrän avulla (Swink 2008, 121).

ADSR on musiikin säveltämisessä käytetty termi, ja on lyhenne sanoista Attack, Decay, Sustain ja Release. Se kuvaa äänen eri vaiheita alusta loppuun. Yhdessä ne muodostavat ADSR-verhokäyrän. Attack on äänen ensimmäinen vaihe; aika joka äänellä kestää nousta nollasta kaikkein kovimpaan tilaansa. Decay kuvaa aikaa, jolloin ääni vaimenee Attackin saavuttamasta kovimmasta äänestä ääneen, joka on tarkoitus säilyttää. Sustain on äänen taso, jonka aikana ääni on tasaisin ja kuulostaa samalta. Release on se aika, joka ääneltä kuluu, kun se vaimenee Sustainin tasaisesta äänestä takaisin hiljaisuuteen (Benediktsson 2011).

Esimerkiksi kitaran kieltä soitettaessa Attack on nopea ja korkea käyrä. Decay on myös nopea mutta tasainen, Sustain pysyy hetken ja Release on pitkä verrattuna muihin vai-

heisiin. Urkuja soitettaessa taas Attack ja Release ovat hyvin nopeat, Decay olematon ja Sustain pitkä ja tasainen. Kuvassa 6 on havainnollistettu edellä kuvatun kitaran ADSR-verhokäyrää.



KUVA 6. Kitaran kielen ADSR-verhokäyrä

Tätä ADSR-verhokäyrää voidaan soveltaa myös pelinkehityksessä arvoja muutettaessa. Pelihahmo voi esimerkiksi pyrähtää juoksuun (nopea, korkea Attack), hidastaa tasaisempaan tahtiin (Decay), säilyttää tämän tasaisen liikenopeuden (Sustain) ja lopuksi pysähtyä (Release). Näitä vaiheita venyttämällä ja muuttamalla voidaan saada kyseinen pelin tilaa muuttava liike tai vastaava tuntumaan erilaiselta ja voidaan vaikuttaa pelituntumaan.

Animaatiossa tätä kutsutaan ”tweenaukseksi”, joka tulee englannin kielen sanasta ”in-betweening”. Termillä tarkoitetaan animaattorin piirtämien liikkeen pääkuvien, ”key-framien”, väliin piirrettäviä kuvia, jotka tekevät liikkeestä sulavamman. Peliohjelmoinnissa tätä voidaan käyttää muuttamaan arvo alkupisteestä loppupisteeseen tiettyä nopeutta tai rataa seuraamalla. (Jonasson & Purho 2012).

Sirius Ballsissa pelaajan mailalla on pieni kiihdytys ennen kuin se saavuttaa tasaisen liikenopeutensa. Kun pelaaja päästää irti ohjaamiseen käytettävästä näppäimestä, pelihahmo pysähtyy välittömästi. Pallot liikkuvat aina samalla nopeudella, ellei niitä kimmoteta erikseen Shift-näppäimellä. Tällöin pallon nopeus vaihtuu suoraan nopeammaksi; se ei kasva luonnollisesti. Lisäksi pallo hidastuu tavalliseen tahtiinsa osuessaan mihin tahansa. Sirius Ballsissa mailan pysähtymistä olisi voinut hidastaa tasaisesti samaan malliin mitä sen lähtiessä liikkeelle.

4.4 ”Context” eli konteksti

Konteksti kuvaa kappaleessa 3.1.2 esitellyn simuloidun tilan vaikutusta pelituntumaan. Tämä simuloitu tila on pelimaailma itsessään, johon kuuluvat pelimaailman objektien vuorovaikutus pelaajan ja ympäristön muiden objektien kanssa ja pelin kenttäsuunnittelu. Konteksti voidaan jakaa kolmeen osaan, jotka kaikki keskittyvät entistä pienempään ja henkilökohtaisempaan osaan pelimaailmaa: korkean, keskikokoisen ja matalan tason kontekstiin (Swink 2008, 139).

4.4.1 Korkea taso

Korkeimmalla kontekstin tasolla keskitytään pelimaailman kokoon, tilaan ja siinä liikkumisen nopeuteen. Jos pelimaailma on laaja ja avoin, se kannustaa pelaajaa tutkimaan ympäristöään. Jos pelimaailma taas on pieni ja ahtaampi, on myös pelikin yleensä keskittynyt enemmän ohjaamaan pelaajaa pisteestä toiseen tarkasti suunnitellulla reitillä (Swink 2008, 141).

Kehittäjän on hyvä pitää mielessään pelinsä tavoite ja mitä hän haluaa sillä sanoa valitessaan peliympäristönsä skaalaa: kauhupeli on tehokkaampi pienessä ja ahtaassa tilassa, missä vastustajilta on hankalampi paeta, kuin laajassa, avoimessa maailmassa. Myös pelin genre, kamerakulma (hahmon silmistä, kolmannen persoonan kamera, isometrinen), kontrollit ja pelimekaniikat on hyvä pitää mielessä peliympäristön skaalaa päätettäessä.

Pelihahmon ja muiden pelimaailman objektien liikkuminen ja sen nopeus voi vaikuttaa pelin pelituntumaan. Esimerkiksi nopeat ja välittömät liikkeet vahvistavat kuvaa ketterästä pelihahmosta. Jos tämä ulkoasultaan ketterä pelihahmo sen sijaan liikkuisi hitaasti ja kömpelösti, se tuntuisi väärältä ja sopimattomalta. Pelin kehittäjällä on vapaat kädet pelinsä nopeuden yksiköihin; peleillä ei ole virallisia, standardisoituja nopeuden yksikköjä. Tämän lisäksi pelaajaa voidaan tavallaan huijata efekteillä, joilla voidaan vahvistaa nopeuden tuntua (Swink 2008, 143).

4.4.2 Keskimmäinen taso

Keskimmäisellä tasolla keskitytään pelihahmon välittömään ympäristöön ja miten pelihahmo on vuorovaikutuksessa ympäristönsä kanssa. Tällä tasolla ei keskitytä pelihahmon nopeuteen numeroina, vaan siihen miten se näkyy ympäristössä. Ympäristön objektien lukumäärä, koko ja luonne sekä niiden sijainti pelimaailmassa ja etäisyys toisistaan - kaikki kenttäsuunnittelun tärkeitä osia - ovat myös osa keskimmäistä tasoa (Swink 2008, 145).

Kirjassaan *Video Game Design: Principles and Practices from the Ground Up* Michael Salmond kuvaa kenttäsuunnittelun tavoitteena olevan saada pelaaja tuntemaan jotakin pelkästä ympäristöstään. Kehittäjän haluttua tunnetta voidaan pyrkiä välittämään pelaajalle esimerkiksi sijoittamalla pelihahmo ahtaaseen, klaustrofobiseen tilaan tai suunnittelemalla reitti, jonka läpi pelaaja tulee mahdollisesti kulkemaan ja sijoittamalla sen varrelle piilotettuja aarteita tai muita peliympäristön tutkimista kannustavaa kerättävää.

Esimerkiksi laaja, avoin pelto on täysin erilainen ympäristö kuin ahdas metsä tai varastorakennus. Pelihahmolla on tilaa liikkua ja pelimaailman objektit voidaan sijoittaa hyvinkin erilleen toisistaan. Ympäristössä liikkuminen on rauhallisempaa ja tavallaan automaattista, kun pelaajan ei aina täydy väistää jotain tai kiertää jonkin ohi. Tällä tavalla pelihahmon ympäristöllä voidaan vaikuttaa pelaajan tämänhetkiseen tunteeseen pelistä.

4.4.3 Matala taso ja konteksti Sirius Ballsissa

Kontekstin alimmalla tasolla keskitytään pelihahmon ja pelimaailman muiden objektien keskinäiseen, henkilökohtaiseen vuorovaikutukseen. Esimerkiksi objektien väliset törmäykset voivat vaikuttaa pelituntumaan. Autopelissä törmäys seinään voi toimia useillakin eri tavoilla riippuen pelin realismista ja tyylistä. Realistisessa autopelissä törmäys seinään voi vakavasti vahingoittaa autoa ja hyvässä pelituntumassa myös tuntuu kovalta törmäykseltä. Sen sijaan nettiroolipelissä pelihahmojen ja objektien törmäyksillä ei yleensä ole mitään vaikutusta toisiinsa: hahmo ei vain enää liiku eteenpäin (Swink 2008, 148).

Tätä tunnetta voidaan vahvistaa luvussa 4.5 esitellyillä viimeistelyn keinoilla, kuten erityisellä animaatiolla, ääniefektillä, ohjaimen tärinällä tai ravisuttamalla pelin kameraa. Tavoitteena on saada pelaaja tuntemaan objektien välinen törmäys tai vuorovaikutus ja saada se tuntumaan ”oikealta”. Pelaajan aiemmat kokemukset ja olettamukset on joko otettava huomioon tai kierrettävä ja esitettävä uudet säännöt, jotka pelaaja oppii ja hyväksyy, jotta pelin konteksti saadaan toimimaan.

Sirius Ballsin pelimaailma on hyvin rajattu ja pieni, minkä tarkoituksena on keskittää pelaajan huomio pelin tapahtumiin. Kentän vasen ja oikea puoli on rajattu koko kentän korkuisilla, läpäisemättömillä palikoilla, jotka kimmottavat pallon takaisin. Pallot liikkuvat samalla nopeudella ja hajotessaan palikat levittävät pienempiä palikoita ympäriinsä. Kenttien suunnittelussa keskityttiin erikoispalikoiden sijoitteluun ja kenttien rakenteeseen: siihen, miten kentän palikat muodostavat yhden kokonaisuuden.

Pelaajien ampumat pallot kimpoavat palikoista ja pelaajan mailasta niiden osumiskulman mukaan. Pallot ja palikat tärisevät törmätessään toisiinsa. Kilpapelinä kenttäsuunnittelun tarkoituksena ei tässä tapauksessa ollut saada pelaajaa tuntemaan mitään erityistä tunnetta, joten kentät suunniteltiin pelattavuuden ja vaihtelun kannalta.

4.5 ”Polish” eli viimeistely

Viimeistely pitää sisällään kaikki pelituntumaan vaikuttavat efektit (visuaaliset, ääniefektit jne.), jotka vahvistavat peliobjektien välistä vuorovaikutusta. Nämä ovat ylimääräisiä efektejä, jotka eivät vaikuta pelin fysiikkaan tai sen sisäisiin laskutoimituksiin. Peli toimisi samalla tavalla ilman niitä, mutta ilman niitä peli toisaalta tuntuisi huomattavasti tyhjemmältä (Swink 2008, 151).

Visuaaliselta puolelta viimeistelyefektejä ovat esimerkiksi partikkeliefektit, osumaa vahvistavat efektit, animaatiot, ruudun tärisyttäminen ja värien vaihtaminen. Ääniefekteihin kuuluvat esimerkiksi auton jarrut, osuman kolahdus tai maalia korostava äänitehoste. Joissain tapauksissa nämä efektit voivat myös yhdessä muodostaa kokonaisuuden, kuten osuman äänitehoste ja visuaalinen efekti yhdessä.

Yksi näistä visuaalisista efekteistä on ruudun ravistaminen, niin kutsuttu ”screen shake”. Tässä pelin kameraa heilutetaan tarkoituksena vahvistaa osuman tai räjähdyksen tai muun tapahtuman vaikutusta. Liitteessä 3 on annettu pala koodia Sirius Ballsista, jossa toteutetaan pelissä käytetty ruudun ravistusefekti. Opinnäytetyön kappaleessa 4.3 esitelty ”tweenaus” on myös yksi viimeistelyn tehokeinoista.

Sirius Ballsissa on useita visuaalisia ja ääniefektejä. Koko ruutu tärisee aina kun pallo osuu mihin tahansa, palikat halkeilevat ja hajotessaan räjähtävät lukuisiksi pienemmiksi palasiksi, pelin tausta vaihtaa väriä, kun jompikumpi pelaaja tekee maalin ja erikoispalikoiden tuhoutuessa ne jättävät jälkeensä kuvan, joka kertoo palikan vaikutuksen. Pallon osuminen, palikoiden hajoaminen ja maali aiheuttavat omat ääniefektinsä.

Näitä efektejä olisi voinut parantaa tietyllä hallinnalla. Tällä hetkellä pelin efektit ovat ajoittain melko radikaaleja, mikä voi haitata pelaajien pelikokemusta. Rajoittamalla joitakin näistä efekteistä pelistä oltaisiin voitu tehdä miellyttävämpi ja selkeämpi katsella.

4.6 ”Metaphor” eli metafora

Metafora voidaan jakaa kahteen osaan, ”representation” eli esittelyyn ja ”treatment” eli käsittelyyn. Esittely on pelin idea; se yhdistää pelihahmon, pelimaailman ja kaikkien pelimaailman objektien ideat. Käsittely on pelin ulkoasun, ääni- ja visuaalisten efektien ja musiikin muodostama yhtenäinen kokonaisuus. Peli voi toimia ilman kumpaakaan näistä, mutta silloin jäljellä on kasa harmaita palikoita tyhjässä, äänettömässä ympäristössä (Swink 2008, 171).

Pelituntuman kannalta metafora on tärkeä, koska se valmistaa pelaajan odottamaan tiettyjä asioita ja olettamuksia peliltä. Mitä enemmän vaste, konteksti ja viimeistely vastaavat pelin esittämää metaforaa, sitä yhtenäisemmältä ja paremmalta peli tuntuu. Kehittäjän on tässä oltava tarkka siitä, etteivät nämä kolme osaa jää juuri vajaaksi pelaajan olettamuksesta. Jos kyseessä on realismiin tähtäävä peli, jonka hahmojen animaatio on vain hiukankin väärässä, se voi helposti tuhota pelaajan immersion pelistä (Swink 2008, 172).

Sirius Ballsin pelimaailma on melko abstrakti, mutta pallot kimpoilevat silti palikoista tarpeeksi oikeissa kulmissa, ettei se riko pelaajan ymmärrystä pelimaailmasta. Pallot eivät myöskään ole pehmeitä, vaan ne osuvat lujaa kentän palikoihin ja kimpoavat tämän kimmoisuuden mukaan.

4.7 ”Rules” eli säännöt

Säännöt yhdistävät pelin ominaisuuksia tai pelimaailman objekteja muodostaakseen jonkin tavoitteen pelaajalle ja ohjatakseen hänen toimintaansa. Näitä voivat olla ”kerää tarpeeksi tähtiä avataksesi seuraavan kentän” tai ”tee 5 maalia voittaaksesi pelin”. Säännöt voidaan jakaa kontekstin tapaan eri tason sääntöihin: korkean, keskikokoisen ja matalan tason sääntöihin (Swink 2008, 180).

Korkealla tasolla säännöt keskittyvät ohjaamaan pelaajan huomion pelin tiettyihin mekaniikkoihin. Nämä säännöt ohjaavat pelaajaa huomioimaan esimerkiksi Sirius Ballsin palikoita. Keskitason säännöt antavat tarkoituksen toiminnalle. Sirius Ballsissa tämä olisi palikoiden ja maalien antamat pisteet. Esimerkki matalan tason säännöstä on vastustajan kestävyys. Vastustajan päihittämiseen käytettävä mekaniikka voi olla sama vastustajasta riippumatta, mutta vastustajan kestävyys erottaa sen kumppaneistaan ja lisää peliin vaihtelua.

Sirius Ballsin sääntöjä ovat palikoiden kestävyys, voittoon vaadittavien pisteiden ja maalien määrä, palikoista ja maaleista saatavat pisteet ja erikoispalikoiden antamat lisäkyvyt ja -voimat.

5 POHDINTA

Opinnäytetyössä tutustuttiin pelituntumaan käsitteenä ja pelisuunnittelun muihin, opinnäytetyön aihetta sivuaviin osiin. Tavoitteena oli selvittää, mitä pelituntuma on ja millaisin keinoin sitä voidaan mitata ja parantaa. Opinnäytetyön valmistuminen viivästyi reilulla kuukaudella arvioidusta ajasta.

Pelituntuma aiheena oli odotettua laajempi, mutta se oli myös kiinnostava. Terminä ja käsitteenä pelituntuma on myös melko subjektiivinen: yhden pelaajan mielestä hyvältä tuntuva peli voi toisen mielestä tuntua hitaalta tai kömpelöltä. Pelituntumalle kuitenkin löydettiin määritelmä, jossa pelituntuma jaettiin kolmeen osaan: reaaliaikaiseen kontrolliin, simuloituun tilaan ja viimeistelyyn.

Pelituntuman parantamiseen on olemassa myös selkeitä, objektiivisesti hyödyllisiä keinoja. Näitä ovat syöte, vaste, konteksti, metafora ja säännöt. Näistä keinoista voisi kirjoittaa huomattavastikin lisää: pelkästään viimeistelyn eri keinoista voisi kirjoittaa useita sivuja, minkä lisäksi opinnäytetyöraportin muutamaa lukua voisi jatkaa ja täydentää pelinkehityksen ulkopuolisistakin aiheista, kuten pelaajien psykologisista reaktioista pelin esittämiin tilanteisiin ja haasteisiin.

Sirius Balls toimi enimmäkseen hyvänä esimerkkinä pelituntuman parantamiselle. Joitakin kohtia, kuten kenttäsuunnittelua, kontekstia ja metaforaa tarkasteltaessa peli ei kuitenkaan tarjonnut kovin monia mahdollisuuksia pelituntuman mittaamisen tai parantamisen tutkimiseen.

Opinnäytetyössä käytetyt materiaalit ovat joko alan ammattilaisten tekemiä tai heidän haastatteluita, mutta aiheen subjektiivisuuden takia käytettävissä ei ollut kovin paljon teoriaa tai faktana hyväksyttyä materiaalia. Tulen hyödyntämään opinnäytetyössä oppimiani pelituntuman parantamiseen käytössä olevia keinoja tulevissa projekteissani.

LÄHTEET

Lähde

Benediktsson, B. 2011. An Introduction to ADSR
https://music.tutsplus.com/tutorials/an-introduction-to-adsr--audio-11150

Swink, S. 2008. Game Feel: A Game Designer's guide to Virtual Sensation. Elsevier

Salmond, M. 2016. Video Game Design: Principles and Practices from the Ground Up.

Computer Hope, 2017. How many keys are on a keyboard?
https://www.computerhope.com/issues/ch001598.htm

Wilde, T. 2016. How WASD became the standard PC control scheme. PCGamer 2016.
https://www.pcgamer.com/how-wasd-became-the-standard-pc-control-scheme/

Jonasson, M. & Purho, P. 2012. Juice it or lose it.
https://www.youtube.com/watch?v=Fy0aCDmgnxg

Nijman, J. W. 2013. Vlambeer – "The art of screenshake".
https://www.youtube.com/watch?v=AJdEqssNZ-U

Brown, M. 2015. The Secrets of Game Feel and Juice
https://schoolofgamedesign.com/project/secrets-of-game-feel-and-juice/

Varney, A. 2008. Jonathan Blow's Shifting Intention
http://www.escapistmagazine.com/articles/view/video-games/issues/issue_163/5146-Jonathan-Blow-s-Shifting-Intention.3

LIITTEET

Liite 1. Linkki Sirius Ballsin tuotesivulle

https://siriusb.itch.io/sirius-balls

Liite 2. Sirius Ballsin palikoiden liikuttamiseen käytettyä koodia

1 (2)

```
using UnityEngine;
using System.Collections;

public class MoveBlockToPosition : MonoBehaviour {

    private Vector3 original;
    private Vector3 distant;
    private Vector3 passed;

    private float speed;
    private float startTime;
    private float journeyLength;
    private float journeyLengthTwo;
    [SerializeField]
    private float ease;
    [SerializeField]
    private float easeTimer;
    private float easeSpeed;
    [SerializeField]
    private float nstart;
    [SerializeField]
    private bool wentPast;

    // Use this for initialization
    void Start () {
        original = transform.position;
        distant = transform.position + new Vector3(0, 0, 15);
        passed = transform.position + new Vector3(0, 0, -1);
        transform.position = distant;

        startTime = Time.time;
        journeyLength = Vector3.Distance(distant, original);
        journeyLengthTwo = Vector3.Distance(passed, original);
        speed = 3.5f * Random.Range(0.85f, 1.25f);
        easeTimer = 0.25f;
        easeSpeed = 0.3f;
        ease = 0.0f;

        nstart = 0.0f;

        wentPast = false;
    }

            // Update is called once per frame
            void Update () {
        if (easeTimer < 1.25f)
        {
            easeTimer += Time.deltaTime * easeSpeed;
            ease = Mathf.Lerp(0.2f, 2.0f, easeTimer);
        }
```

2 (2)

```
        float distCovered = (Time.time - startTime) * (speed * ease);
        float fracJourney = distCovered / journeyLength;
        transform.position = Vector3.Lerp(distant, passed, fracJourney);

        if (transform.position == passed && wentPast == false)
        {
            wentPast = true;
            easeTimer = 0.8f;
            ease = 0.0f;
            distCovered = 0.0f;
            fracJourney = 0.0f;
        }

        if (wentPast == true)
        {
            startTime = 0.0f;
            if (easeTimer < 1)
            {
                easeTimer += Time.deltaTime * easeSpeed;
                ease = Mathf.Lerp(1f, 1.2f, easeTimer);
            }

            distCovered = (nstart - startTime) * (speed * ease);
            fracJourney = distCovered / journeyLengthTwo;
            transform.position = Vector3.Lerp(passed, original, fracJourney);

            nstart += Time.deltaTime;
        }

        if (wentPast == true && transform.position == original)
        {
            gameObject.GetComponent<MoveBlockToPosition>().enabled = false;
        }
    }
}
```

Liite 3. Ruudun ravistamiseen käytettyä koodia Sirius Ballsista

1 (2)

```
using UnityEngine;
using System.Collections;

public class ShakeCamera : MonoBehaviour {

            private float duration;
            private float speed;
            private float magnitude;
            private bool test;

            // Use this for initialization
            void Start () {
//                      duration = 0.3f;
//                      speed = 6.0f;
//                      magnitude = 1.2f;
                        test = false;
            }

            public void SetValues(float dur, float spd, float mag) {
                        duration = dur;
                        speed = spd;
                        magnitude = mag;
            }
```

2 (2)

```
        public void PlayShake() {
                StopAllCoroutines ();
                StartCoroutine ("Shake");
        }

        // Update is called once per frame
        void Update () {
                if (test) {
                        test = false;
                        PlayShake ();
                }
        }

        IEnumerator Shake() {
                float elapsed = 0.0f;
                Vector3 originalCamPos = transform.position;
                float randomStart = Random.Range (-1000.0f, 1000.0f);

                while (elapsed < duration) {
                        elapsed += Time.deltaTime;

                        float percenComplete = elapsed / duration;

                        float damper = 1.0f - Mathf.Clamp (2.0f *
                        percenComplete - 1.0f, 0.0f, 1.0f);

                        float alpha = randomStart + speed * percen-
                        Complete;

                        float x = Mathf.PerlinNoise (alpha * 2.0f -
                        1.0f, 0.0f) * 2 - 1;
                        float y = Mathf.PerlinNoise (0.0f, alpha *
                        2.0f - 1.0f) * 2 - 1;

                        x *= magnitude * damper;
                        y *= magnitude * damper;

                        transform.position = new Vector3 (x, y,
                        transform.position.z);

                        yield return null;
                }

                transform.position = originalCamPos;
        }
}
```